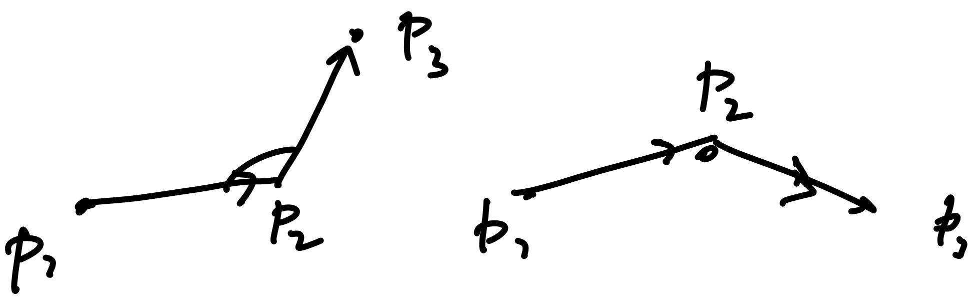
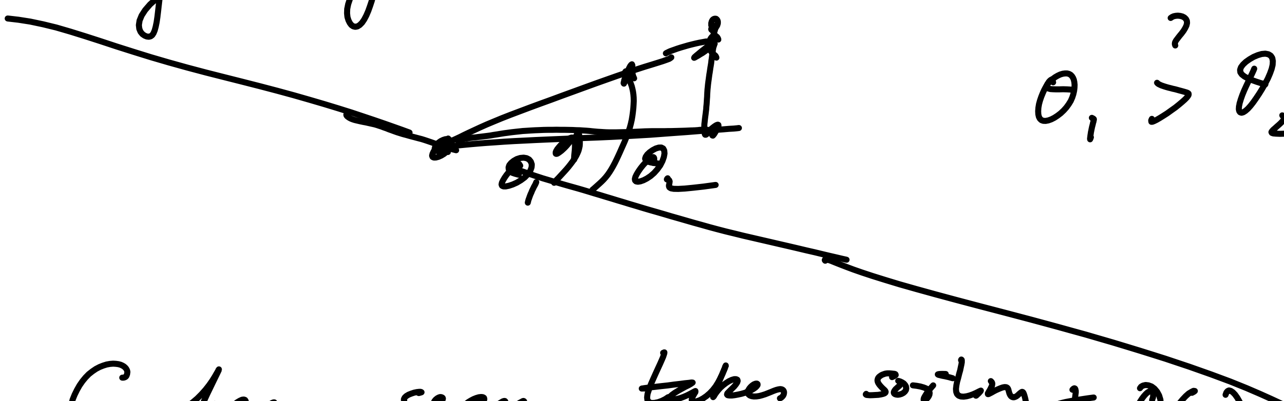
COL 702 Lecture 24 Nov 2, 2017



Area of Δ is given a determinant and the left/right turn is given by sign



$\theta_1 \overset{?}{>} \theta_2$

Graham scan takes sorting + $O(n)$ time

Jarvis march " $O(h\,n)$ steps

h = # boundary vertices

The best possible balance of input/output can be achieved as $O(n \log h)$

"Output sensitive" algorithms

Can we do better than $O(n \log n)$?

Consider sorting of a given set S of n numbers $x_1, x_2 \ldots x_n$

$S \longrightarrow S'$ $\quad (x_1, x_1^2)\ (x_2, x_2^2) \ldots (x_n, x_n^2)$

$CH(S')$

All points of S' will be on the boundary

By projecting back to x axis we can deduce the sorted set of points in S

Total time for sorting S : $\underbrace{O(n)}_{\text{initial transfo}} + T(n)$ ↗ time for CH

$+ O(n)$ reverse transform

S can be sorted in $O(n) + T(n)$ time

$Sort(n) = \Omega(n \log n)$

Therefore $O(n) + T(n)$ must be $\Omega(n \log n)$

$\Rightarrow$ $T(n)$ must be $\Omega(n \log n)$

Lower bound for sorting is $\Omega(n \log n)$

In general for problems π_1 and π_2

we say π_1 is reducible to π_2

$\pi_1 \leq \pi_2$ | $\pi_1 \propto \pi_2$

if any algorithm for π_2 can be used to also solve π_1

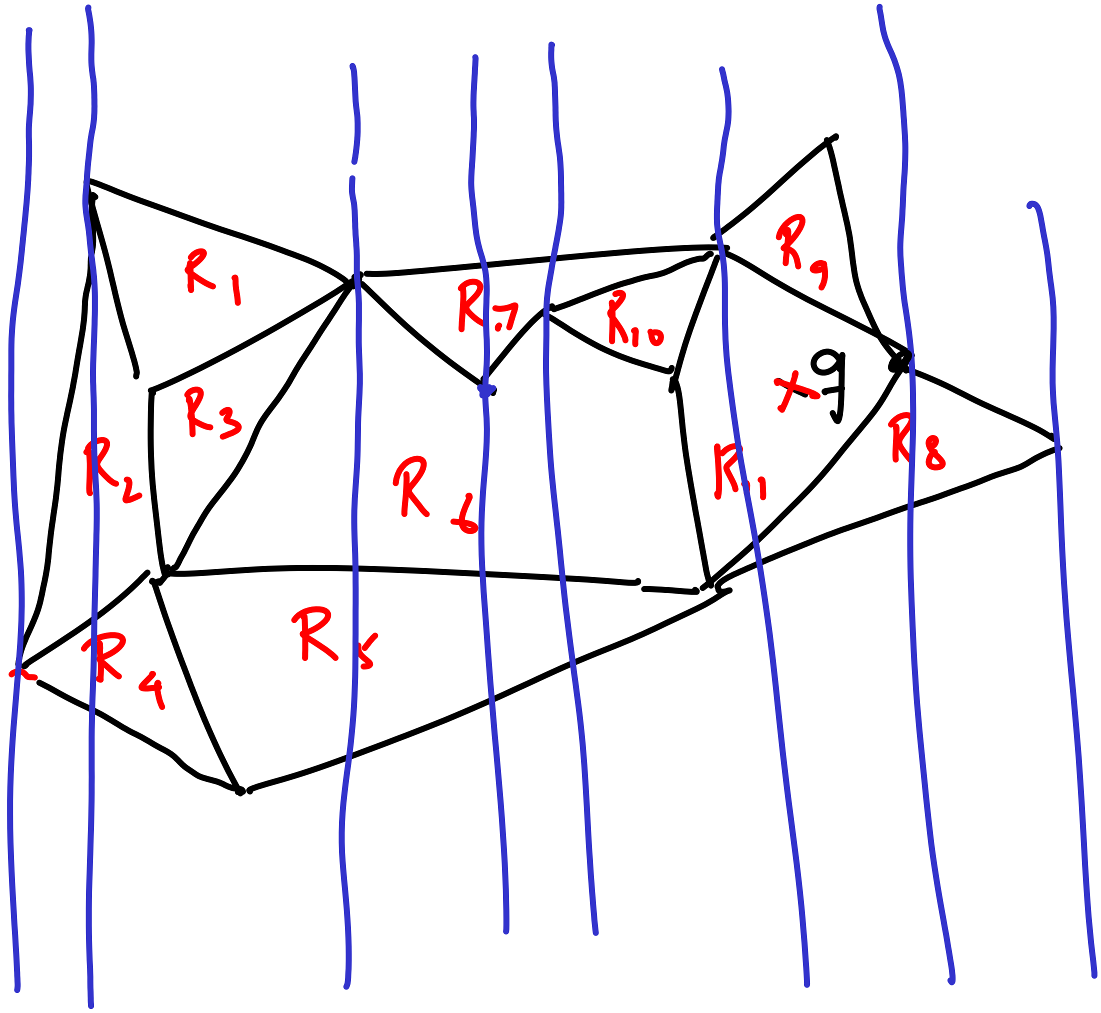
Observation 1. An upperbound for π_2 implies upper bound for π_1

2. Any lower bound for π_1 implies a similar lower bound for π_2

We showed Sorting $\leq$ Convex hull

The # boundary faces in d-dim convex hull can be $n^{\lfloor d/2 \rfloor}$

(upper bound theorem)



Binary search in the x direction will tell us the vertical strip containing the query location $q = (x^*, y^*)$

By doing a binary search within the vertical strip, we can locate the query

Total query time : $O(\log n)$

What about the data structure ?

The total size of the data structure for the second phase is

$n_1 + n_2 + n_3 \cdot \qquad n_k$

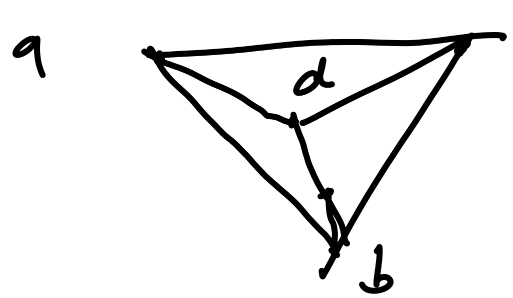
where n_i is the # of segment in slab i

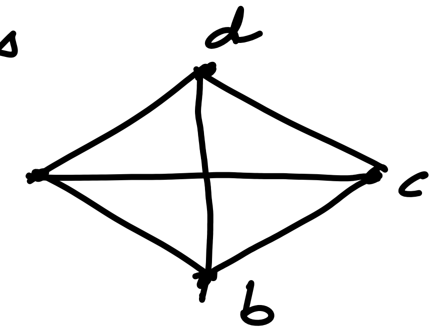
This can be $\Omega(n^2)$ for worst case inputs

PLANAR POINT LOCATION

A planar map can be thought of as a planar graph

A graph that can be drawn on the plane with edge crossings





e : # edges v = # vertices f : # faces

$$v - e + f = 2 \quad (1)$$ Euler's relation

The maximum # faces in a planar graph is given by the Δted region of a planar graph

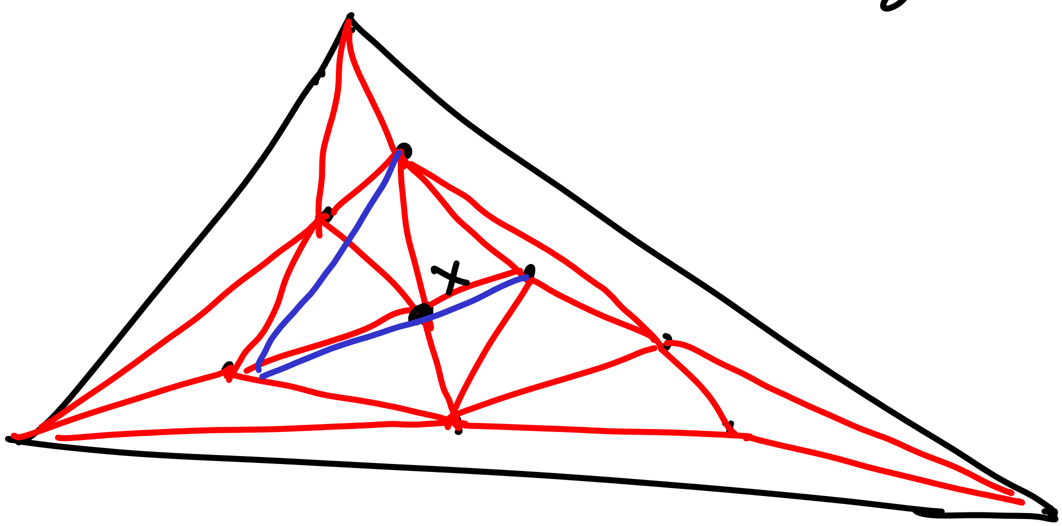
In a Δted planar region

$$2e = 3f \Rightarrow f = \frac{2}{3}e$$

Subst in (1) $v - e + \frac{2}{3}e = 2$

$$\boxed{e = 3(v-2) < 3v} \quad (2)$$

wlog, let us consider a planar subdivision where every face (including the outer face) is a triangle



Step 1 Eliminate "many" non-adjacent vertices

Step 2 Re triangulate some of these regions

Step 3 Keep track of how the new Δs intersect with the previous (eliminated) Δ's

$$\begin{array}{cccc} T_1^0 & T_2^0 & T_3^0 & T_4^0 \\ T_1^n & T_2^n & T_3^n & \end{array}$$

Repeat steps 1 & 3 until # vertices is less than 25

For point location queries we first find the location with respect to the 25 point graph, perhaps by brute force

We refine the search over the different levels we have created till we know the location in the original graph.